

ELEC 204 Digital Design Preliminary Lab Report

Preliminary Lab XX

Name: Student Name

Date: date of submission on blackboard (MM/DD/YYYY)

*Please delete the highlighted lines and write your own parts for the report.

**Reminder: Your lab grade is a weighted average of your performance before, during and after the lab:

Total lab grade = Preliminary work* (30%) + Lab interview and demo* (40%) + Lab report (30%)

*****Please make sure the preliminary report does not exceed 10 A4 pages.**

For the preliminary work, please provide here:

- *Your answers to the questions in the tutorial lab*
- Circuit schematics (if applicable)
- State diagram (only applies for Labs 3, 4 and 5)

ELEC 204 Digital Design Lab Report

Lab 01

Name: Barış KAPLAN, Id Number: 69054

Date: date of submission on blackboard (03/06/2021, 6 March 2021)

*Please delete the highlighted lines and write your own parts for the report.

**Reminder: Your lab grade is a weighted average of your performance before, during and after the lab:

Total lab grade = Preliminary work* (30%) + Lab interview and demo* (40%) + Lab report (30%)

***Please make sure the lab report does not exceed 10 A4 pages.

****Please make sure you indicate the name of your lab collaborator (if there is any) who you worked together to solve the lab questions. Do not change your lab collaborator throughout the semester.

1. Introduction and objectives

Explain the objectives of the lab (refer to the lab instruction sheet),

Lab1's objectives

- 1-) Getting used to the Xilinx ISE software platform.
- 2-) Learning how to create new projects in the platform which is called "Xilinx 64 Bit Project Navigator".
- 3-) Learning how to implement the XOR (exclusive OR) gate using the VHDL programming language.
- 4-) Learning how to simulate the VHDL code.

Explain what your code has to do and describe how you did it.

My code has to be the implementation of the XOR gate using the intermediate signals. In my code, I should reach the expanded form of 'X XOR Y' by defining and using the intermediate signals which make the final expanded form of 'X XOR Y' in the Lab1Code.vhd. Moreover, when X or Y is 1, my code's output value, which is the value of Z, should be 1. Otherwise, my code's output value should be 0. In my code, I have defined, named, and used intermediate signals. Firstly, I have used the "not Y" signal. Secondly, I have used "(not Y) and X" signal. Thirdly, I have used "not X" signal. Then, I have used "(not X) and Y" signal. In the end, I have used "((not Y) and X) or ((not X) and Y)" signal, which corresponds to the expanded form of (X XOR Y). In the signal definition and naming part, I have defined 4 intermediate signals which correspond to "not Y", "(not Y) and X", "not X", and "(not X) and Y" signals.

2. Methods

Explain the inputs (how many bits, names of the inputs),

INPUTS

In my VHDL code, there are two inputs. The names of the inputs are X and Y. X is one bit. Y is one bit. These two inputs correspond to two bits, which means that there are two bits in total for the inputs.

Explain the outputs (how many bits, names of the outputs),

OUTPUTS

In my VHDL code, there is one output. The name of the output is Z. This output corresponds to one bit, which means that there is one bit in total for the outputs.

Explain what the VHDL code must do

WHAT THE VHDL CODE MUST DO

When the value of X is 1 and the value of Y is 0, the VHDL code of the XOR gate should give 1 as the value of Z. When the value of X is 0 and the value of Y is 1, the VHDL code of the XOR gate should give 1 as the value of Z. When the value of X is 1 and the value of Y is 1, the VHDL code of the XOR gate should give 0 as the value of Z. When the value of X is 0 and the value of Y is 0, the VHDL code of the XOR gate should give 0 as the value of Z. (X and Y are inputs. Z is the output.) The VHDL code should reach the expanded form of 'X XOR Y' as the output signal by using the intermediate signals.

Explain how your code works

HOW MY CODE WORKS

For X=0 and Y=0, my code gives 0 as the output. For X=0 and Y=1, my code gives 1 as the output. For X=1 and Y=0, my code gives 1 as the output. For X=1 and Y=1, my code gives 0 as the output. When the simulation ends, I observe that the output values, which are the values of Z, are as expected. I can also observe these output values using the expanded form of 'X XOR Y' which I found as the output signal in my VHDL code. When I put necessary values in the expanded form, I can observe that the outcomes are as expected also in this case. My code's intermediate signals correspond to the small parts in the expanded form of 'X XOR Y'. These intermediate signals are for 'Not X', 'Not Y', '(Not Y) and X', and '(Not X) and Y'. The output signal of my code is "((not Y) and X) or ((not X) and Y)".

Provide the truth table

THE TRUTH TABLE OF XOR

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

3. Problems encountered, errors and warnings resolved

Explain what problems you encountered while writing your code.

While I am trying to write my code, I encountered with some errors related to the missing semi-colons. Moreover, I have encountered with some errors which are related to undeclared variables in the code.

Explain what synthesis errors and warnings you observed.

I have observed synthesis errors related to the missing semi-colons and undeclared variables. I have not observed any synthesis warnings.

4. Conclusion

Provide a 1 paragraph summary of the lab and explain what you learned from this lab.

In the lab1, firstly, I create the project. Secondly, I set the programming language to VHDL. Then, I create the VHDL Module, and name the inputs and outputs of my code. The names of the inputs are X and Y. The name of the output is Z. After that, I modify the content of the VHDL code that comes when I create the VHDL module and open it. Subsequently, I create a VHDL Test Bench. I modify the Lab1Sim.vhd. I run the simulation and observe the output values which are based on the changing values of X and Y. I see that when X or Y is 1, the output value is 1. Otherwise, the output value is 0. From this lab, I have learned how to implement the XOR gate. Moreover, I have learned that the expanded form of XOR is ((not X) and Y) or ((not Y) and X). In addition to that, I have learned how to create a new project, how to perform the simulation, how to create a VHDL module, and how to create a VHDL Test Bench in 'Xilinx 64 Bit Project Navigator'. I have learned how to observe the output values, when the input values change. Finally, I have learned that when X=0 and Y=0, Z will be 0; when X=0 and Y=1, Z will be 1; when X=1 and Y=0; Z will be 1, and when X=1 and Y=1, Z will be 0.

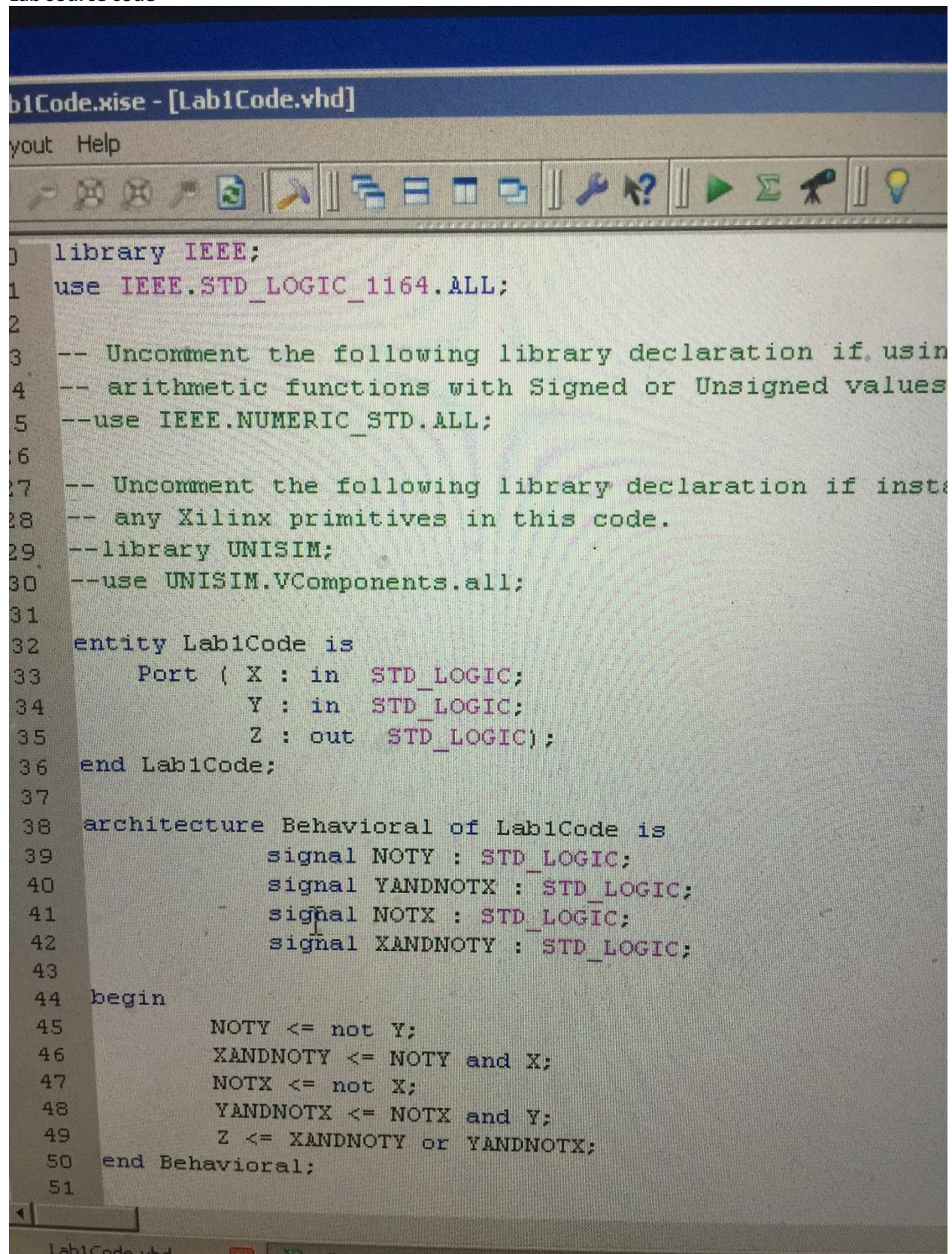
References

1. Please cite any resource (web site, book, youtube video) you used for this lab.

Website: https://www.electronics-tutorials.ws/logic/logic_7.html

Youtube video: <https://youtu.be/yweEa7toGwo>

Lab source code



```
Lab1Code.xise - [Lab1Code.vhd]
yout Help

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity Lab1Code is
    Port ( X : in  STD_LOGIC;
           Y : in  STD_LOGIC;
           Z : out STD_LOGIC);
end Lab1Code;

architecture Behavioral of Lab1Code is
    signal NOTY : STD_LOGIC;
    signal YANDNOTX : STD_LOGIC;
    signal NOTX : STD_LOGIC;
    signal XANDNOTY : STD_LOGIC;
begin
    NOTY <= not Y;
    XANDNOTY <= NOTY and X;
    NOTX <= not X;
    YANDNOTX <= NOTX and Y;
    Z <= XANDNOTY or YANDNOTX;
end Behavioral;
```